

---

# Exploiting KML and Atom/geoRSS As A Metadata Standard



**Brian Wilson and Zhangfan Xing**  
Jet Propulsion Laboratory





# Outline

- KML/GEarth is a scalable publishing, search, and visualization platform
- OpenSearch (REST) protocol, with Geo Extensions
  - Everything is search, and all results are feeds (Atom or KML).
  - See <http://www.opensearch.org/>
  - Atom & KML results should be equivalent (round-trip convert)
- Opportunistic Use of Atom & KML for Metadata
  - Embedded metadata fields
  - Micro XML formats: georss:box (spatial bounding box)
- Need more embeddable micro-formats standardized
  - timeSpan: in KML, but not standard in opensearch/atom
  - Links to browse image, list of retrieved variables, dataset ontology, domain information, documentation, etc.
- What is killer app for metadata? Search.
  - Modular, micro metadata; versus full ISO19115/11179
  - Need both, meet in the middle





# KML / GEarth Platform

- Scalable publishing & commodity visualization
  - The innovation is this powerful combination.
  - Tell a science story with an accompanying viz.
  - Google crawls and indexes KML files
  - GEarth provides spatial Placemark search, and visual animations over time
- Embed more metadata tags in KML (or Atom) files
  - Georss tags based on GML, different from KML spatial
  - Micro XML formats: georss:bbox (spatial bounding box)
- GEarth provides Placemark (KML) search
  - Viewport defines implicit lat/lon rectangle
  - Then do keyword search into indexed KML files
  - But what about time spans or other metadata?
- Opportunity
  - Exploit scalable publishing & search
  - Add more metadata to the combination





# OpenSearch Protocol

- Everything is search, all results are feeds (or KML).
- REST search interface
  - [http://your.music.com/-/albums?q=lady+madonna  
&startIndex=1&count=200&format=atom](http://your.music.com/-/albums?q=lady+madonna&startIndex=1&count=200&format=atom)
  - Returns Atom feed listing first 200 albums that satisfy query keywords
  - Metadata embedded in XML feed entries
- Protocol
  - Feed header contains links to description doc. for server
  - Also contains links to previous, next, & last batch of results
  - Opensearch aggregators can display results in HTML (using stylesheet) and auto-traverse result set
  - Google Data extends opensearch with more conventions, used pervasively in GApps and GFeedReader



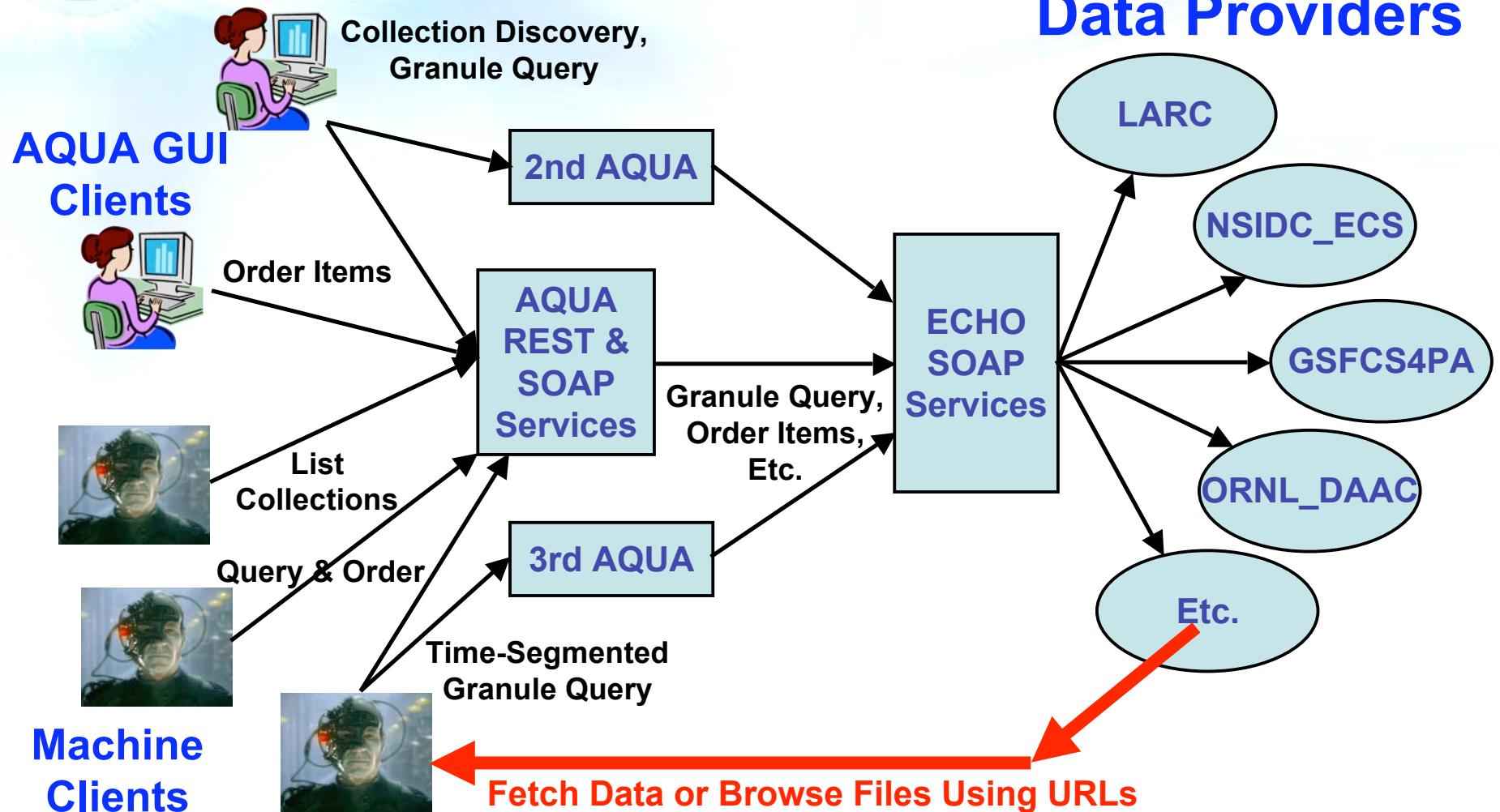


# GeoRSS

- GeoRSS = Geographically Encoded Objects for RSS
  - Tags to embed in RSS or Atom feeds
- GeoRSS-Simple
  - **<georss:point>45.256 -71.92</georss:point>**
  - Assumes WGS84 lat/lon coordinates in decimal degrees
  - Also line, box, and polygon
  - Feature & relationship tags
- GeoRSS-GML
  - **<georss:where>**  
**<gml:Point> <gml:pos>45.256 -71.92</gml:pos></gml:Point>**  
**</georss:where>**
  - A GML Profile (subset)
- Time not yet standardized.



# AQUA Client Architecture





# Interfaces, Interfaces, Interfaces

- You can never be too simple, or have too many interfaces.
- **AJAX GUI for Humans**
  - Also visible interface for marketing
- **Machine-callable SOAP services**
  - For large-scale automation, human out of the loop
  - GeoRegionQuery, OrderItems, etc.
- **Equivalent REST interface for services**
  - rest.py?\_service=aqua&\_method=GeoRegionQuery
- **OpenSearch (REST) interface**
  - Everything is search, and all results are feeds.
  - See opensearch.org





# AQUA Open Search Interface

## ■ Discover a Collection

- `http://server/aqua//collections?q=water+vapor&startIndex=1-&count=200&format=atom`
- Returns Atom feed listing collections satisfying query keywords
- Metadata included in XML feed

## ■ Space/Time Query for Granules

- `http://server/aqua/-/granules/providerId/datasetShortName?time=2006-01-01T00:00:00,2006-02-01T00:00:00&bbox=south,west,north,east&responseGroups=Large&startIndex=1&count=200&format=kml`
- Returns Atom feed (or KML document!)
- Granule metadata (time, georss:box) and URL's included

## ■ OpenSearch (GData) Features

- GoogleData standard uses and extends OpenSearch
- Search aggregators auto-handle Atom feed, traverse result sets
- Ingest data feed into Google Earth or Google Apps



# AQUA Space/Time Query Result

```
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:georss="http://www.w3.org/2003/01/geo/wgs84_pos#"
      xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/">
  <title>OpenSearch space/time granule query results for provider GSFCS4</title>
  <updated>2008-07-08T18:16:30.508269</updated>
  <author><name>AQUA ECHO Client</name></author>
  <id>uri:http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-01-01T00:00:00,2006-02-01T00:00:00</id>
  <opensearch:totalResults>510</opensearch:totalResults>
  <opensearch:startIndex>1</opensearch:startIndex>
  <opensearch:itemsPerPage>200</opensearch:itemsPerPage>
  <opensearch:Query role="request" searchTerms="???" startIndex="1"/>
  <fubar:time>2006-01-01T00:00:00,2006-02-01T00:00:00</fubar:time>
  <georss:box>0.,0.,60.,179.</georss:box>
  <link href="http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-01-01T00:00:00,2006-02-01T00:00:00" type="application/atom+xml" rel="self"/>
  <link href="http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-01-01T00:00:00,2006-02-01T00:00:00" type="application/vnd.google-earth.kml+xml" rel="alternate"/>
  <link href="http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-01-01T00:00:00,2006-02-01T00:00:00" type="text/html" rel="alternate"/>
  <link href="http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-01-01T00:00:00,2006-02-01T00:00:00" type="application/atom+xml" rel="previous"/>
  <link href="http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-02-01T00:00:00,2006-03-01T00:00:00" type="application/atom+xml" rel="next"/>
  <link href="http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-01-01T00:00:00,2006-02-01T00:00:00" type="application/atom+xml" rel="first"/>
  <link href="http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-03-01T00:00:00,2006-04-01T00:00:00" type="application/atom+xml" rel="last"/>
  <link href="http://sciflo.jpl.nasa.gov/aqua/-/granules/GSFCS4PA/AIRX2RET?t=2006-01-01T00:00:00,2006-02-01T00:00:00" type="application/opensearchdescription+xml" rel="search"/>
<entry>
```



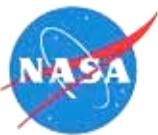


# AQUA Space/Time Query Result (2)

```
<entry>
  <title>SC:MYD10_L2.004:9936826</title>
  <link href="urn:SC:MYD10_L2.004:9936826"/>
  <link rel="data" href="urn:SC:MYD10_L2.004:9936826"/>
  <link rel="dap" href="http://ecs.nasa.gov/dap/modis/-"/>
  <link rel="browse" href="http://browse.echo.nasa.gov/-
    NSIDC_ECS/2006/01/04/:BR:Browse.001:9936830:1.BINARY"/>
  <id>urn:SC:MYD10_L2.004:9936826</id>
  <updated>2006-03-02T13:04:48.0000</updated>
  <fubar:time>2006-01-01T10:30:00,2006-01-01T10:35:00</fubar:time>
  <georss:box>south west north east</georss:box>
  <content type="xml">
</entry>
</feed>
```

- Granule metadata includes:

- Time range
- Bounding box
- Link to browse image
- DAP URL pointing to data file (if possible), otherwise ftp





# AQUA Space/Time Query Result (3)

```
<fubar:time>2006-01-01T10:30:00,2006-01-01T10:35:00</fubar:time>
<georss:box>west south east north</georss:box>
<content type="xml">
  <GranuleURMetaData xmlns="http://echo.nasa.gov/echo/v10">
    <ECHOItemId>G82210940-NSIDC_ECS</ECHOItemId>
    <GranuleUR>SC:MYD10_L2.004:9936826</GranuleUR>
    <InsertTime>2006-01-03 07:08:10.0000</InsertTime>
    <LastUpdate>2006-03-02 13:04:48.0000</LastUpdate>
    <ECHOInsertDate>2006-01-04 09:21:27.0000</ECHOInsertDate>
    <ECHOLastUpdate>2006-03-04 03:43:34.0000</ECHOLastUpdate>
    <Orderable>Y</Orderable>
    <CatalogItemId>G82210940-NSIDC_ECS</CatalogItemId>
    <CollectionMetaData>
      <ShortName>MYD10_L2</ShortName>
      <VersionID>4</VersionID>
      <DataSetId>MODIS/Aqua Snow Cover 5-Min L2 Swath 500m V004</DataSetId>
    </CollectionMetaData>
    <DataGranule>
      <SizeMBDataGranule>33.954171</SizeMBDataGranule>
      <ReprocessingPlanned>further update is anticipated</ReprocessingPlanned>
      <ReprocessingActual>reprocessed</ReprocessingActual>
      <ProducerGranuleID>MYD10_L2.A2006001.1030.004.2006003133830.hdf</ProducerGranuleID>
      <DayNightFlag>DAY</DayNightFlag>
      <ProductionDateTime>2006-01-03 14:00:47.0000</ProductionDateTime>
      <LocalVersionID>SCF V4.0.0</LocalVersionID>
    </DataGranule>
```



# Micro XML formats (tagging)

- Simpler is Better. Reuse and embed. Easy adoption.
  - <georss:box>south west north east</georss:box>
  - <time>start,end</time>
  - Dublin Core metadata
  - Taxonomy of physical domains & retrieved variables from domain & dataset ontologies
  - What else could we embed?
- What is the standard <time> tag?
  - KML: <TimeSpan><begin/><end/></TimeSpan>
  - Simpler: <time>start[,end]</time>
  - Georss-GML considering:

```
<georss:when><gml:TimePeriod gml:id="1">
  <gml:begin>2006-06-17T09:55:00Z</gml:begin>
  <gml:end>2006-06-17T10:30:00Z</gml:end>
</gml:TimePeriod></georss:when>
```

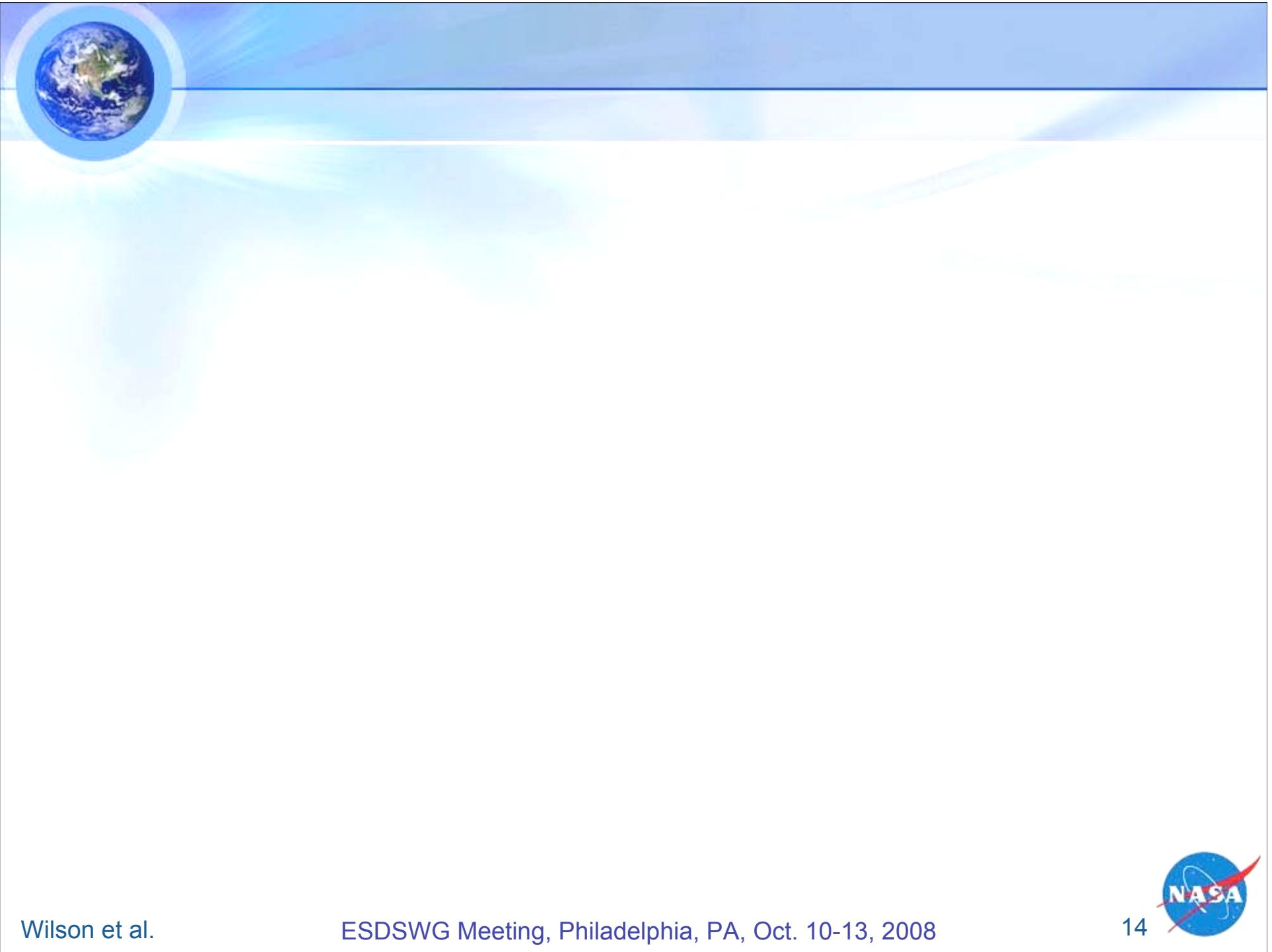




# Conclusions

- Opportunity to reuse KML & Atom as modular metadata standard
  - Killer app, GEarth, already exists: publish, search, viz
  - Embed small XML formats like georss
  - Start small and build up to more metadata
- Opensearch protocol standardizes query result sets
  - Return Atom or KML
  - AQUA search client over ECHO uses it
- What other killer apps are there for metadata?
  - Provenance tracing
  - On demand recomputation of products
- Light vs. heavier weight metadata
  - Full metadata standards, ISO19115/11179
  - Extract modules from ISO standards, like georss from GML
  - Do both, but how do we meet in the middle?







# Data Stewardship Using URIs

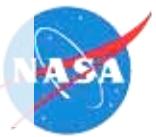


## ■ Query & URI Resolution

- Query & Resolution services could be provided by separate institutions, replicated, or transferred (NASA → NOAA)
- Name Lookup (URIs → URLs) can “hide” entire process of locating & loading objects from tape archive (ordering data)
- URI’s assigned by hierarchical naming authorities
- Example -- AIRS project permanently names its products:  
`us:gov:nasa:eos:AIRS:AIRS.2003.01.02.004.L2.RetStd`

## ■ Enabling Long-Term Climate Science

- Systems must scale to “huge” queries & orders
- Support repeatable science analysis over years of data
- AQUA Client (Automated Query and Access, recent ACCESS ECHO grant, PI: Wilson)





# The Handle System

- Alternative to URI's or URN's

- Open source software, developed by CNRI
- Operational system
- Global name resolver delegates to local resolvers
- Simple, so scalable
- Format: <Local Resolver>/<Permanent Names>

- Possible Handle Scheme

- us.gov.nasa.eos.airs/AIRS.2003.01.02.004.L2.RetStd
- AIRS project runs local name resolver
- Permanent names adapted from unique granule names or ECHO ItemIds
- Resolver translates permanent name to one or more URL's

- **Query → Perm. Names → URL's → Data Access**

- Names → UR's done by local name resolver
- Resolver maintains list of URL's by crawling data archives
- Transfer of data from tape to disk could be hidden inside the Name Resolution service





# Datasets in SciFlo

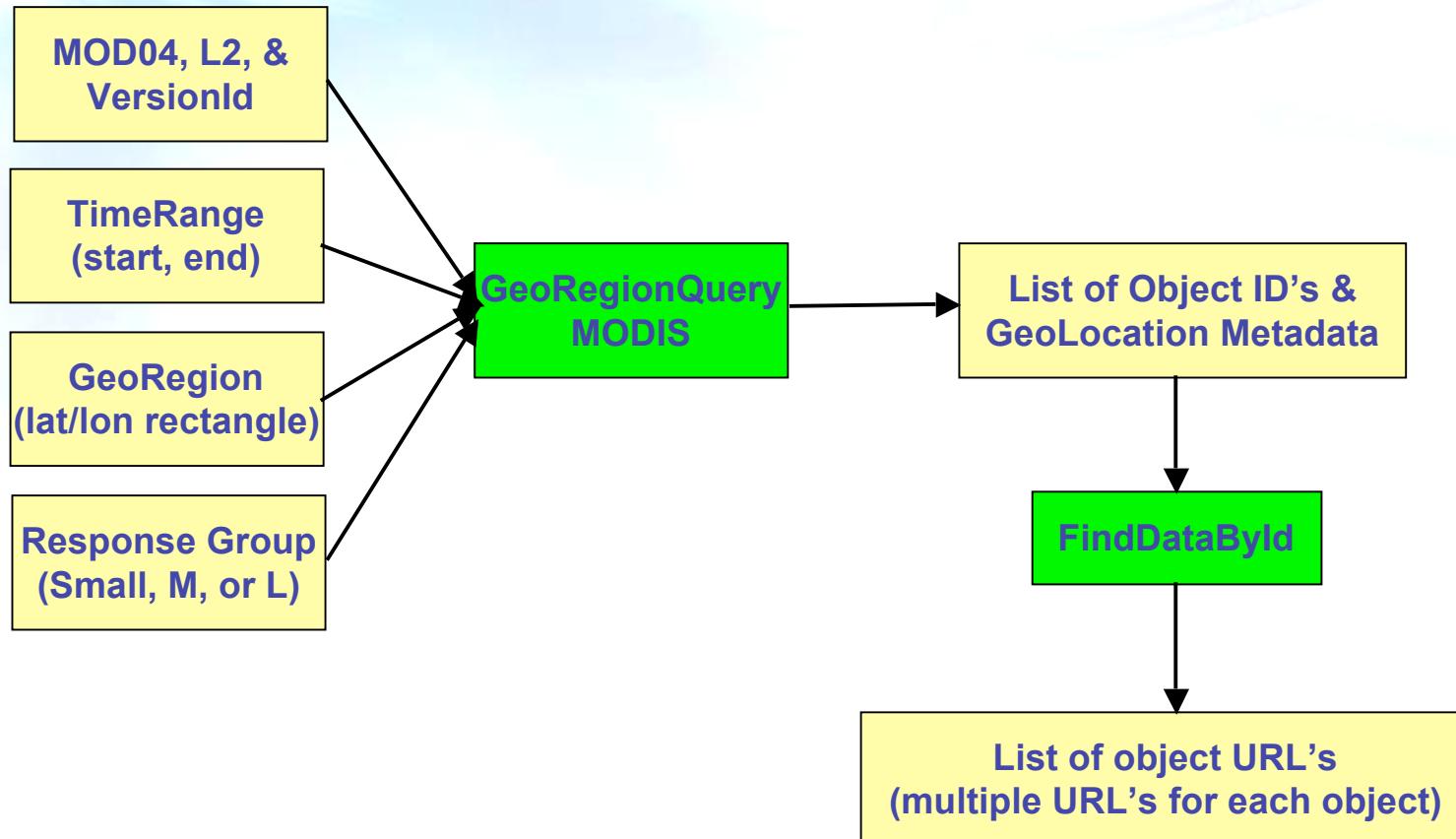
- A SciFlo Dataset is:

- *Specified* as a space/time query over collections of data products (or retrieved physical variables)
  - **GeoRegionQuery**(DataProduct, TimeRange, LatLonRegion)
  - **GeoRegionQuery**(PhysicalVariable, TimeRange, LatLonRegion)
- *Realized* as a list of object ID's or URI's (permanent names)
  - GeoRegionQuery returns unique objectIds along with geolocation metadata
- *Accessed* using a list of URL's pointing to on-line replicas of the data objects (files).
  - **FindDataById**(objectIds) → URLs (ftp, http, or OpenDAP)
  - Translate unique object ID's into list of on-line locations in DataPools or any SciFlo node
  - DataPools & SciFlo P2P network are “crawled” to update distributed translation tables
  - No need to publish presence of data, continuously discovered
  - SciFlo network is a **distributed** cache for scientific datasets



# GeoRegionQuery → Dataset

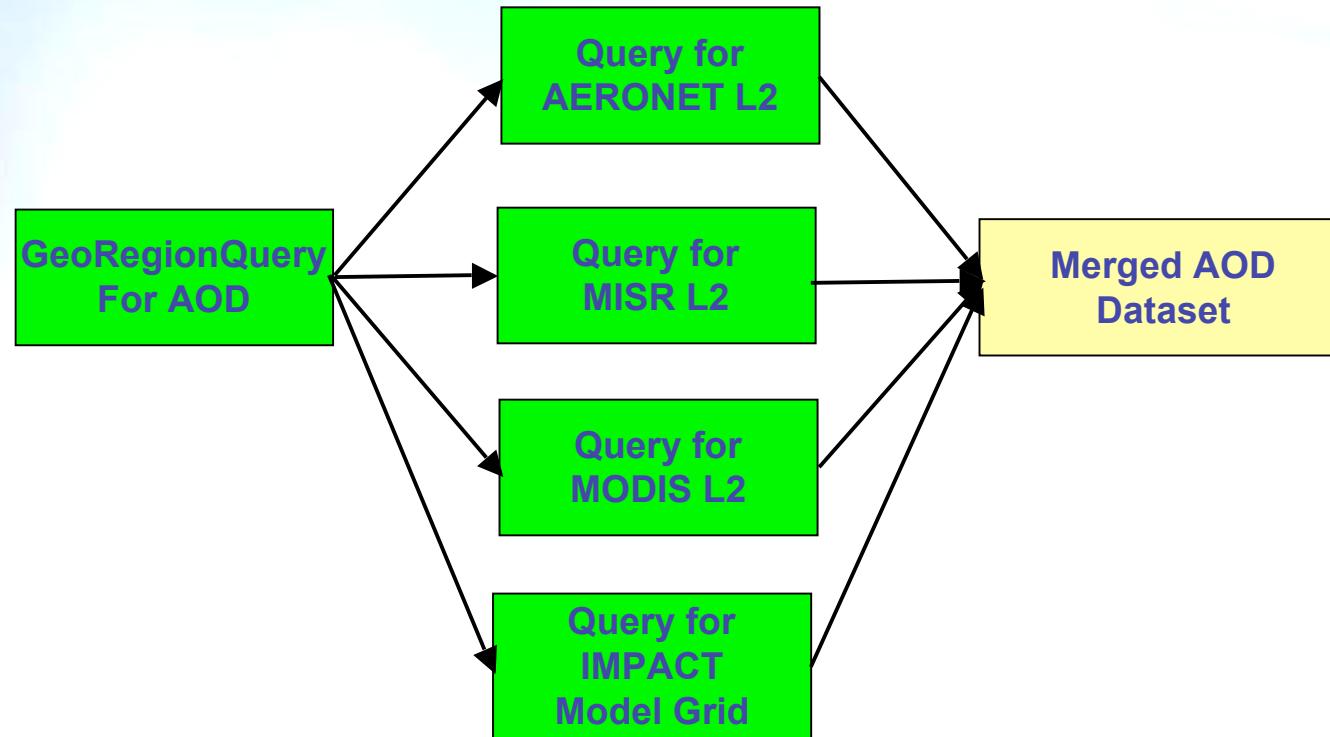
## Space/Time Query for MODIS Dataset





# GeoRegionQuery → Datasets

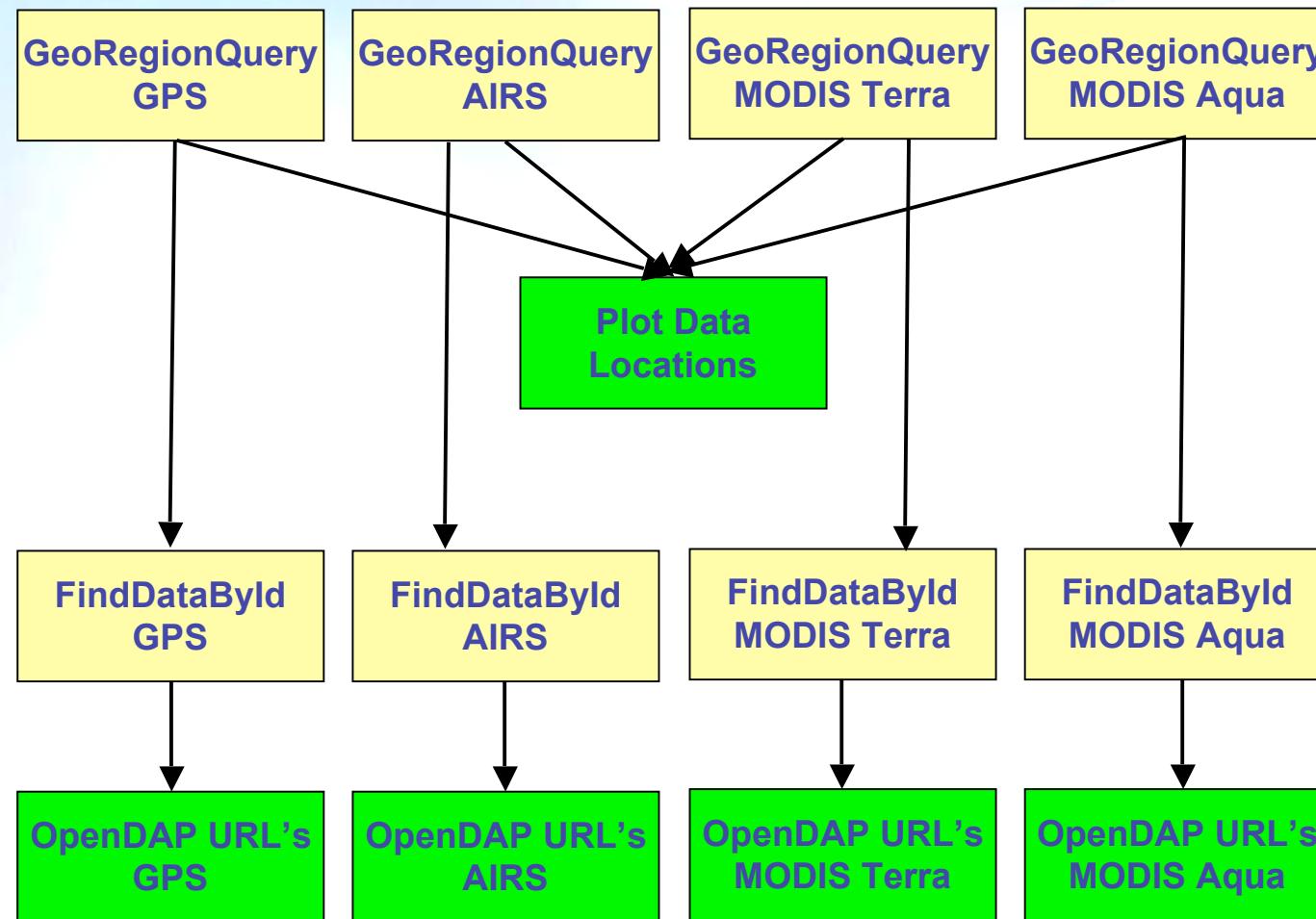
## Query for Aerosol Optical Depth Variable: Match Multiple Instruments/Models





# Example Workflow: Locate 4 Datasets

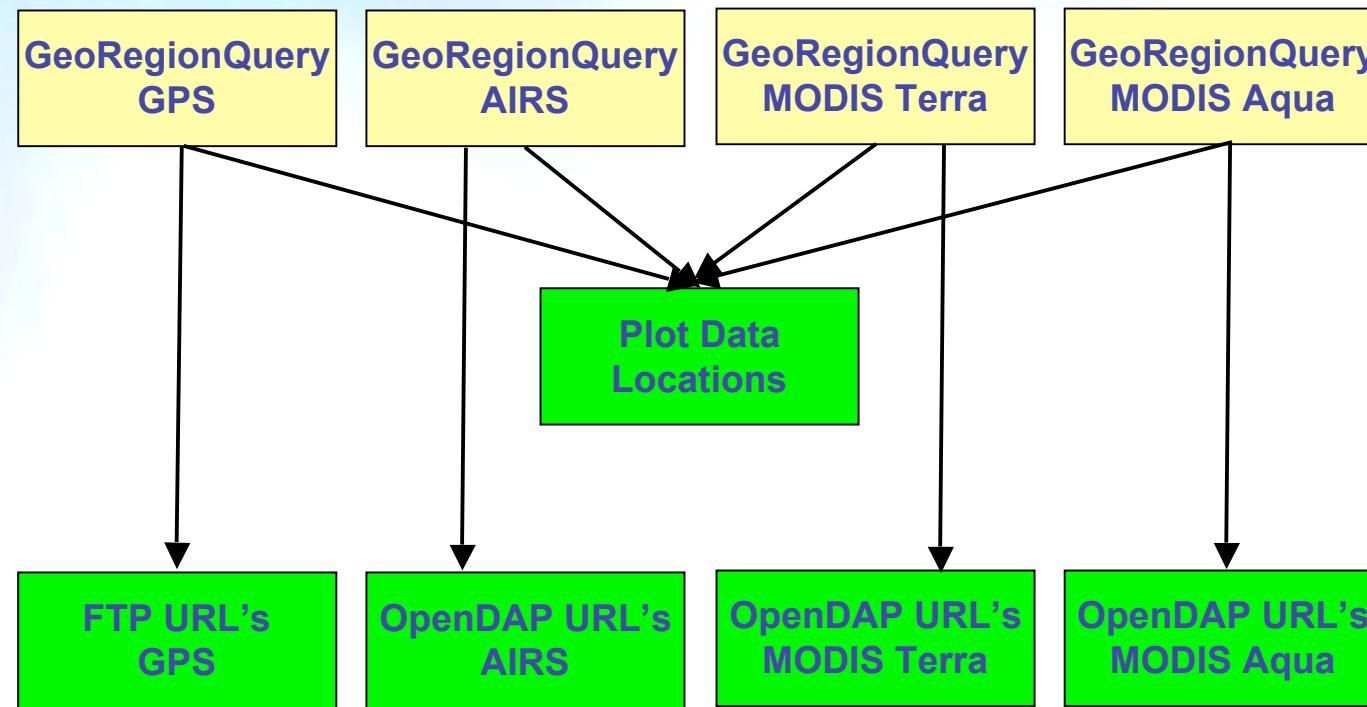
## Space/Time Query for GPS, AIRS, & MODIS





# Example Workflow: Locate 4 Datasets

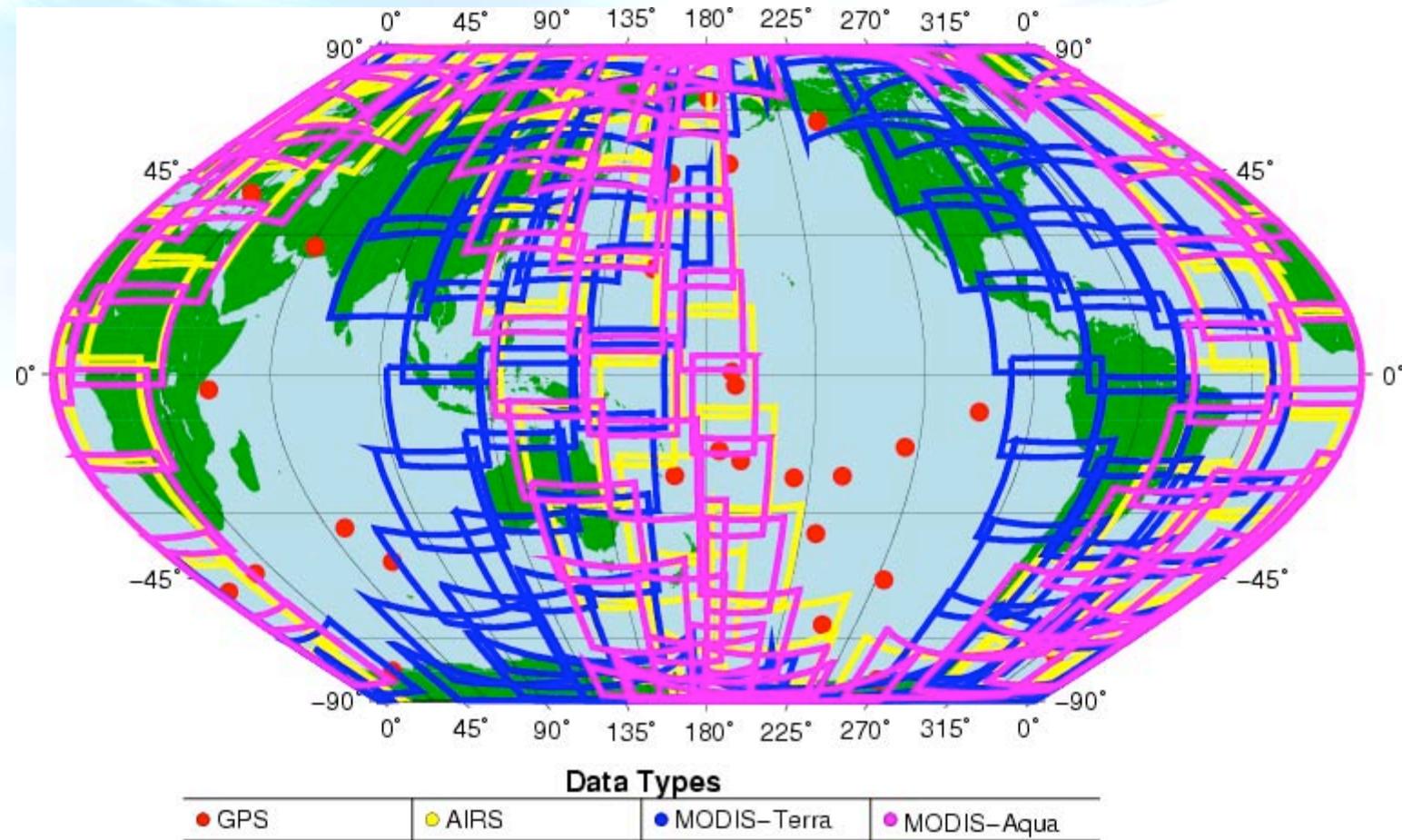
## Space/Time Query for GPS, AIRS, & MODIS





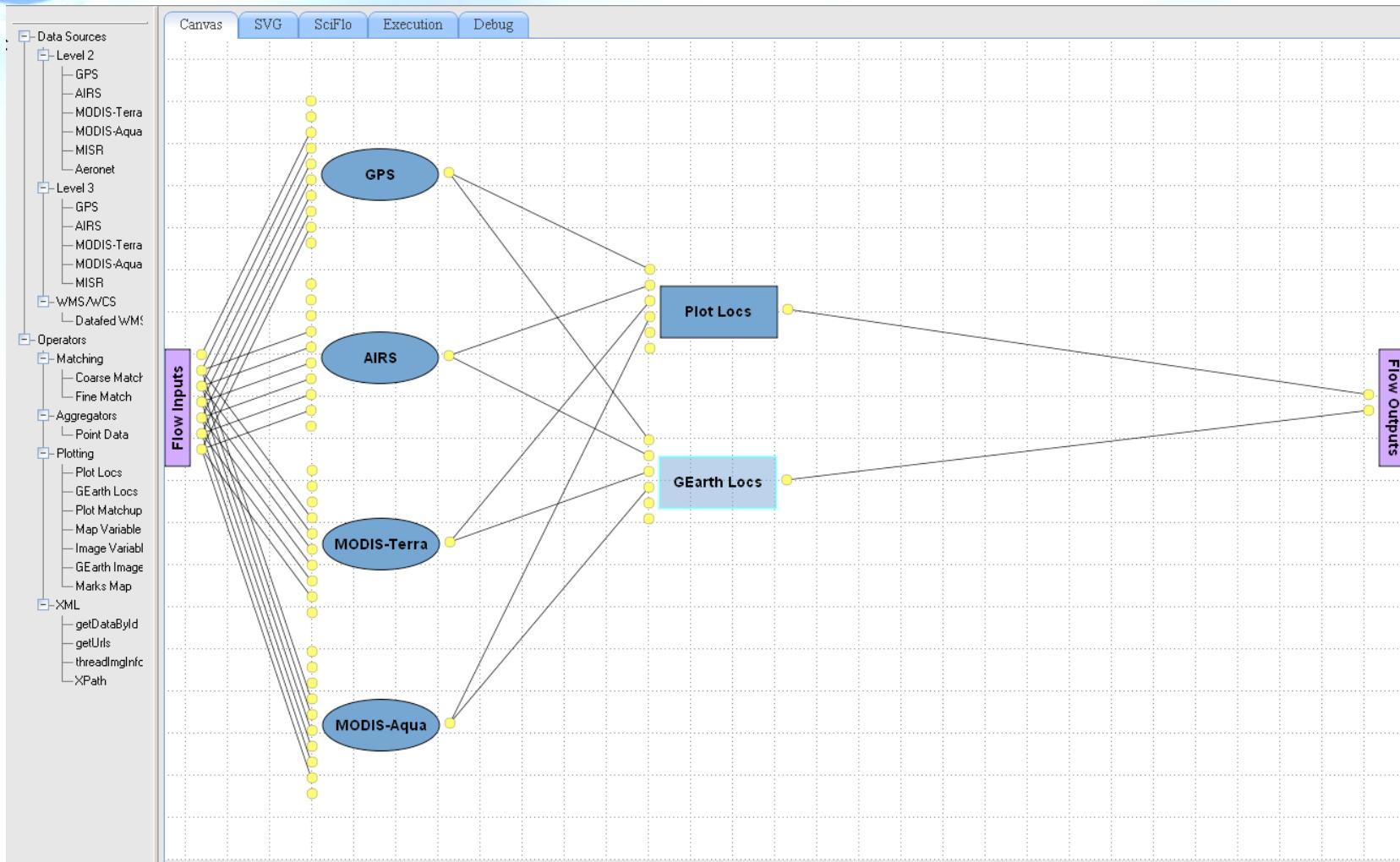
# Plot of Granule Bounding Boxes

## Space/Time Query for GPS, AIRS, & MODIS



# iEarth Flowchart

## Space/Time Query for GPS, AIRS, & MODIS





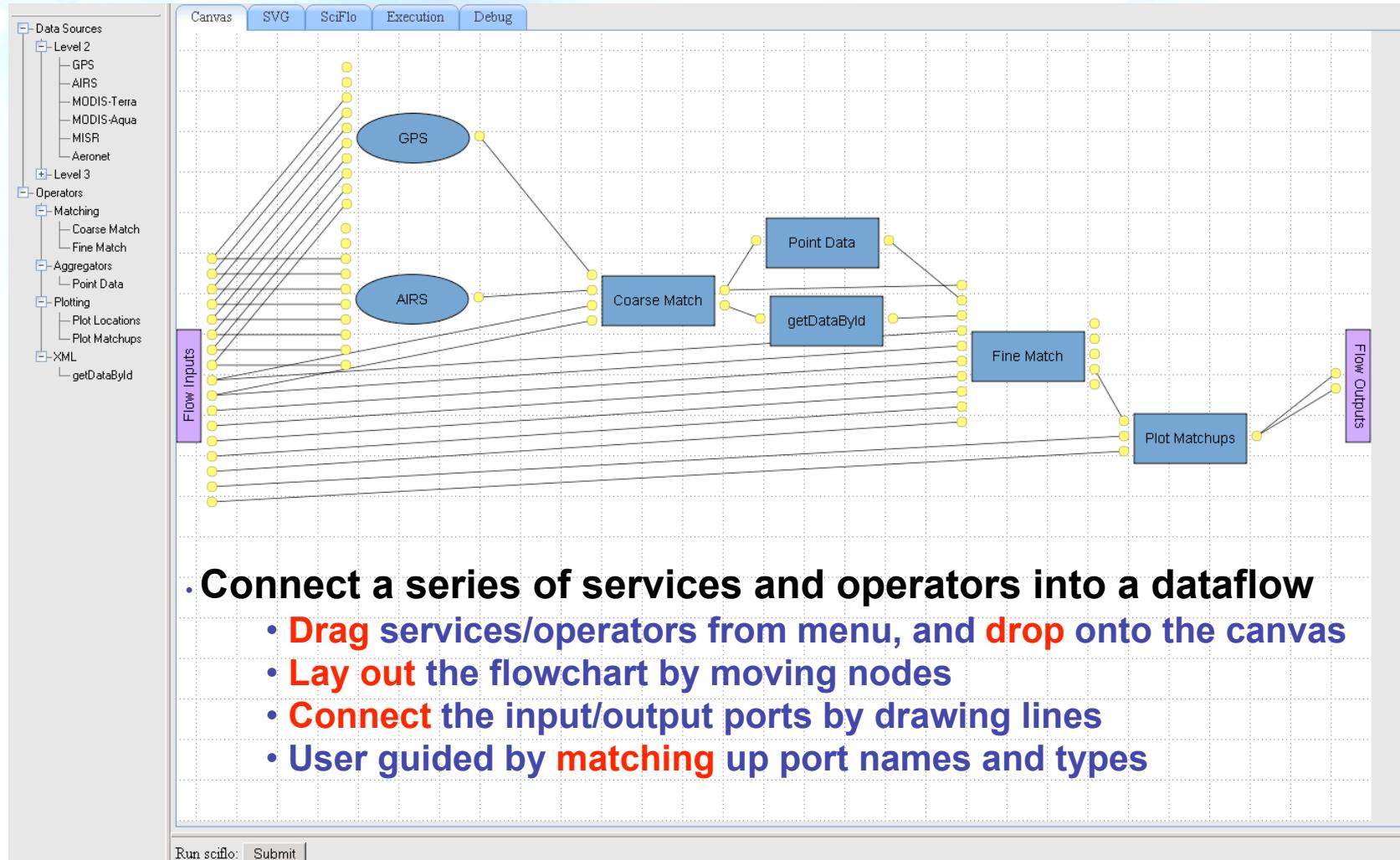
# SciFlo's Innovative Use of IT

- Pervasive use of XML metadata
  - Computers are stupid because we don't give them enough semantic metadata!
  - XML metadata more important than files of numbers
  - XML Microformats – “micro” for easy adoption
  - Ontologies: concept maps / taxonomies for science domains
- Intentional Programming
  - Visual Programming (in your web browser!)
  - Declare the workflow in an XML document; minimize code
  - Programming at a human (conceptual) level
  - Powerful high-level programming language (python)
- Data Stewardship thru Permanent Names
  - Permanent names are underutilized on the Internet
  - Product Query → URIs → URLs → Data Access
  - URI: us:gov:nasa:eos:AIRS:AIRS.2003.01.02.004.L2.RetStd
  - ECHO ItemIds can serve as a form of URI



# Visual Dataflow Programming

## GPS & AIRS Level-2 Space/Time Matchup



- Connect a series of services and operators into a dataflow
  - Drag services/operators from menu, and drop onto the canvas
  - Lay out the flowchart by moving nodes
  - Connect the input/output ports by drawing lines
  - User guided by matching up port names and types

(AJAX Javascript and SVG Programming by Gerald Manipon & Wilson)

